

Lab Assignment 3

Due 2030

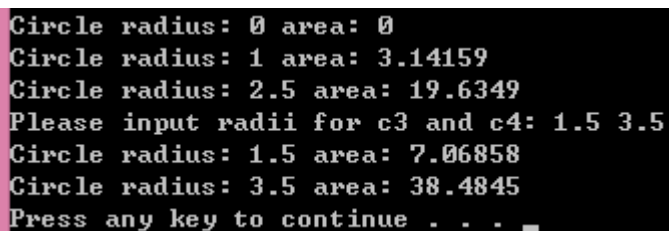
Lab Grading Policy: Attendance 20%, Score 80%

In case you have difficulty in finishing the exercises on time, you should upload them before **noon on Saturday** and a penalty of 20% discount will be applied on your score. No late submission is permitted. We will in general post the reference solutions **by Sunday**.

Exercise 1 (40%): Write a `Circle` struct (`Circle.h` and `Circle.cpp`) so one can set its radius through a constructor, by a member function or from an interactive input. In addition, the `Circle` object can report its radius and area when we print the object. Below are a sample client code and output:

```
#include <iostream>
#include "Circle.h"
using namespace std;

int main()
{
    Circle c1; // default constructor, radius is 0.0
    print (cout, c1);
    c1.setRadius(1);    // This sets c1 radius to 1
    print (cout, c1);
    Circle c2(2.5);    // Constructor that sets c2 radius to 2.5
    print (cout, c2);
    Circle c3, c4; // default constructor, radius is 0.0
    cout << "Please input radii for c3 and c4: ";
    read(read (cin, c3), c4); // c3 radius is 1.5, c4 radius is 3.5
    print(print (cout, c3), c4);
    return 0;
}
```



```
Circle radius: 0 area: 0
Circle radius: 1 area: 3.14159
Circle radius: 2.5 area: 19.6349
Please input radii for c3 and c4: 1.5 3.5
Circle radius: 1.5 area: 7.06858
Circle radius: 3.5 area: 38.4845
Press any key to continue . . .
```

Exercise 2 (40%): Write a `Maze` (迷宫) struct (`Maze.h` and `Maze.cpp`) for simple maze manipulation. A typical layout and syntax for the maze are given below. The first line of the layout file is two values indicating the number of rows and columns in the maze. Each subsequent line contains a single row of the maze. The border of the maze and obstacles within the maze are represented by plus signs. Spaces represent open area within the maze. S

indicates the starting point of the maze and F is the finish. For example, a 6×12 maze layout looks like:

6 12

```

+ + + + + S + + + + + +
+ + + + +   + +   + + +
+               + + +
+   +   +   + +   +
+ + +   + +               F
+ + + + + + + + + + + +

```

Write a constructor to read the maze layout file, a non-member function `printLayout` to print the maze layout, a `find` member function to find a path and a non-member function `printPath` to print the path with the following information: (1) coordinates of the start, e.g., (5, 0), and finish, e.g., (11, 4) for the layout given above and (2) coordinates of a valid path.

(Hint: Implement your `find` member function so you can try a path until you get stuck, then retrace your steps until you can follow another path, and repeat. Eventually, you will either find a path that leads to the destination, or try all paths and decide that no solution exists.)


Below are a sample client code and output:

```

#include <iostream>
#include "Maze.h"
using namespace std;

int main()
{
    Maze m("maze612.txt");
    printLayout(cout, m);
    cout << endl;
    cout << "-----" << endl;
    cout << "Your escaping path:" << endl << endl;
    printPath(cout, m.find());
    return 0;
}

```



```
C:\Windows\system32\cmd.exe
6 12
+++++S+++++
+++++0++ +++
++ 0000+++
++ + ++000+
+++ ++ oF
+++++

-----
Your escaping path:

The path location:
<5, 1>
<5, 2>
<6, 2>
<7, 2>
<8, 2>
<8, 3>
<9, 3>
<10, 3>
<10, 4>
請按任意鍵繼續 . . .
```